Collision Avoidance and Resolution Multiple Access (CARMA)

Rodrigo Garcés and J.J. Garcia-Luna-Aceves
Computer Engineering Department
School of Engineering
University of California
Santa Cruz, CA 95064, USA
E-mail: garces/jj@cse.ucsc.edu

The collision avoidance and resolution multiple access (CARMA) protocol is presented and analyzed. CARMA uses a collision avoidance handshake in which the sender and receiver exchange a request to send (RTS) and a clear to send (CTS) before the sender transmits any data. CARMA is based on carrier sensing, together with collision resolution based on a deterministic tree-splitting algorithm. For analytical purposes, an upper bound is derived for the average number of steps required to resolve collisions of RTSs using the tree-splitting algorithm. This bound is then applied to the computation of the average channel utilization in a fully connected network with a large number of stations. Under light-load conditions, CARMA achieves the same average throughput as multiple access protocols based on RTS/CTS exchange and carrier sensing. It is also shown that, as the arrival rate of RTSs increases, the throughput achieved by CARMA is close to the maximum throughput that any protocol based on collision avoidance (i.e., RTS/CTS exchange) can achieve if the control packets used to acquire the floor are much smaller than the data packet trains sent by the stations. Simulation results validate the simplifying approximations made in the analytical model. Our analysis results indicate that collision resolution makes floor acquisition multiple access much more effective.
Keywords: wireless, packet radio, multiple-access protocol

1. Introduction

Several medium access control (MAC) protocols have been proposed over the past few years that are based on three- or four-way handshake procedures meant to reduce the number of collisions among data packets, thereby providing better performance than the basic ALOHA or CSMA protocols [3,8,12,13]. The concept of "floor acquisition" was introduced by Fullmer and Garcia-Luna-Aceves

[0] This work was supported in part by DARPA under Grants DAAB07-95-C-D157 and DAAH04-96-10210

Report Documentation Page		*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE **1998**	2. REPORT TYPE	3. DATES COVERED **00-00-1998 to 00-00-1998**
4. TITLE AND SUBTITLE **Collision Avoidance and Resolution Multiple Access (CARMA)**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of California at Santa Cruz,Department of Computer Engineering,Santa Cruz,CA,95064**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR’S ACRONYM(S)
		11. SPONSOR/MONITOR’S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release; distribution unlimited**		
13. SUPPLEMENTARY NOTES		
14. ABSTRACT		
15. SUBJECT TERMS		

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**30**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

[5,6] for MAC protocols based on such handshake procedures. In a single-channel network, floor acquisition multiple access (FAMA) entails allowing one and only one station at a time to send data packets without collisions. To achieve this, a station that wishes to send one or multiple data packets must send a request-to-send packet (RTS) to an intended destination and receive a clear-to-send packet (CTS) from it, before it is allowed to transmit any data. The sender uses carrier sensing or packet sensing (i.e., relies on the detection of carrier or the reception of entire error-free packets) to decide when the channel is free. RTSs are required to last a minimum amount of time that is a function of the channel propagation time, and CTSs are required to last longer than an RTS time plus the maximum round-trip time.

A floor acquisition strategy is very attractive in packet-radio networks, because it provides a building block to solve the hidden-terminal problem that arises in CSMA [14]. Variants of this basic strategy can be designed using different types of contention-based MAC protocols like ALOHA or CSMA to transmit RTSs into the channel, and three- or four-way handshakes can be implemented (i.e., RTS-CTS-DATA or RTS-CTS-DATA-ACK). The efficiency of FAMA protocols using carrier sensing to eliminate the hidden-terminal problems of CSMA has been verified and analyzed [6]. However, the throughput of a FAMA protocol still degrades rapidly once stations start retransmitting unsuccessful RTSs that collide repeatedly with other RTSs. This paper shows that using tree-splitting algorithms to resolve the collision of RTSs in a FAMA protocol running in a high-speed network improves substantially the performance of FAMA protocols under high-load conditions. This is because data packets never collide with control packets in a FAMA protocol and the propagation delays and duration of RTSs and CTSs are much smaller than the duration of data packets, which means that the average time needed to resolve the collisions of RTSs is very small compared to the duration of data packets. Therefore, even if every RTS has to go through collision resolution, the channel can still be used for useful data the majority of the time. In contrast, it is well known that tree-splitting algorithms do not provide much improvement for CSMA protocols [2], which stems from the fact that the data packets themselves are used for collision resolution.

Section 2 describes a specific protocol, which we call CARMA (for collision avoidance and resolution multiple access), and which uses non-persistent carrier sensing for the transmission of RTSs and a tree-splitting algorithm to resolve collisions of RTSs. The basic deterministic tree-splitting algorithm used in CARMA is much the same as the methods used in other protocols based on the identities of nodes. Many MAC protocols have been proposed based on collision resolution (e.g., RAMA [1], TRAMA [9], DQRUMA [10], DQRAP [15]). The interesting feature in CARMA is the combination of a known collision resolution algorithm with RTS/CTS exchanges, which provides a stable and very efficient protocol that does not require time slotting or availability of base stations capable of detecting multiple simultaneous transmissions. Section 3 computes the average number of steps required to resolve collisions, differentiating among steps involving collisions, successful transmissions, and idle times, and upper bounds on these averages. These bounds extend prior results on tree-splitting algorithms, which focused on the total number of steps needed for collision resolution [4,15]. The importance of the bounds we present is that they are independent of the number of stations in the network; Section 4 uses them to compute lower bounds of the average throughput achieved by CARMA when a very large population of nodes is assumed. We show that the throughput achieved by CARMA as the arrival rate of RTSs increases is very close to the maximum throughput that can be achieved by any protocol based on collision avoidance (i.e., RTS/CTS exchanges prior to the transmission of data packets) when the propagation delays and control packets are much smaller than the data packets or packet trains sent by stations. Section 5 presents simulation results validating the simplifications used in our analytical model, and show that the bounds obtained analytically are close to the throughput values obtained by simulation. Section 6 offers our concluding remarks.

2. Protocol Description

FAMA protocols solve collisions by backing off and rescheduling RTS transmissions [5,6]. As with CSMA protocols, this procedure yields good results if the RTS traffic is low; however, the probability of collisions increases as the rate of transmissions increases, with a corresponding decrease of system throughput. Eventually, as the RTS transmission rate increases in a FAMA protocol, the constant RTS collisions cause the channel to collapse, bringing the flow of data packets to a halt. A method to stabilize FAMA protocols is, of course, increasing the backoff delays to reduce congestion; however, this simply forces nodes to attempt accessing the channel at low rates that avoid instability. It is well known that a method to stabilize ALOHA and CSMA protocols is by means of collision resolution; however, because data packets are used to resolve collisions, the maximum throughput that can be achieved in a stable CSMA or ALOHA protocol is limited [2]. In contrast, CARMA (for collision avoidance and resolution multiple access) addresses the limitations of FAMA and contention-based protocols using carrier sensing for the transmission of RTSs (with which stations attempt to acquire the floor) and a tree-splitting algorithm to resolve collisions of RTSs. Throughout this paper, we assume a simple tree-splitting algorithm for collision resolution, but more sophisticated algorithms [2] can be used instead. Each station must know the maximum number of stations allowed in the system and the maximum propagation delay in the network.

2.1. Basic CARMA

We begin the description of CARMA assuming that all stations are in line of sight of one another. Each station is assigned a unique identifier, a stack, and two variables ($LowID$ and $HiID$). $LowID$ is initially the lowest ID number that is allowed to send an RTS, while $HiID$ is the highest ID number that is allowed to send an RTS. Together, they constitute the allowed ID interval that can send RTSs, i.e., attempt to acquire the floor. If the ID of a station is not within this interval, it cannot send its RTS. As we describe subsequently, the stack is simply a storage mechanism for ID intervals that are waiting to get permission to send an RTS.

In CARMA a station can be in one of five different states, namely:

- PASSIVE: The station has no local packets pending and no transmissions are detected in the channel.
- RTS: The station is trying to acquire the floor and has sent an RTS.
- XMIT: The station has the floor and is sending data packets.
- REMOTE: The station is receiving transmissions from other stations and has no local packet to send.
- BACKOFF: The station has local packets pending and had to reschedule its request for the floor.

Fig. 1 gives a formal description of CARMA. When a passive station has one or multiple packets to send, it first listens to the channel. If the channel is clear (i.e., no carrier is detected) for one maximum round-trip time, the station transmits an RTS. The sender then waits and listens to the channel for one maximum round-trip time, plus the time needed for the destination to send a CTS. When the originator receives the CTS from the destination, it acquires the floor and begins transmitting its data packet burst. The sender is limited to a maximum number of data packets, after which it must release the channel and must compete for the floor at a later time if it still has data packets to send.

If the sender of an RTS does not receive a CTS within a time limit, the sender and all other stations in the system know that a collision has occurred. As soon as the first collision takes place, every station divides the ID interval $(LowID, HiID)$ into two ID intervals. The first ID interval is $(LowID, \lceil \frac{HiID+LowID}{2} \rceil - 1)$, which we will call the backoff ID interval, while the second ID interval is $(\lceil \frac{HiID+LowID}{2} \rceil, HiID)$ and is called the allowed ID interval. Each station in the system updates the stack by executing a PUSH-stack command, where the key being pushed is the backoff ID interval. After this is done, the station updates $LowID$ and $HiID$ with the values from the allowed ID interval. This procedure is repeated each time a collision is detected.

Only those stations that were in the RTS state at the time the first collision occurred are allowed into the collision-resolution phase of the protocol. All other stations are in REMOTE state or BACKOFF state, until all collisions are resolved; those stations in BACKOFF state when all collisions are resolved must then backoff for a random amount of time. Accordingly, when a passive station obtains a packet to send and the channel is not clear for one maximum round-trip time, the station enters the BACKOFF state.

Collision resolution evolves in terms of collision-resolution intervals of which there are three cases. In the first interval of the collision-resolution phase all stations in the allowed ID interval that are in the RTS state try to retransmit an RTS. If none of the stations within this ID interval request the channel, the channel is idle for a time period T_i, i.e., an idle step occurs. At this point, a new update of the stack and of the variables *LowID* and *HiID* is due. Each station executes a POP-stack command. This new ID interval now becomes the new *HiID* and *LowID*. The same procedure takes place if, during the first collision-resolution interval, only one station is requesting the channel, i.e., a success step occurs. In this case, the originator receives the CTS from the destination and begins transmitting its data packet burst, after which the station releases the channel and transitions to the PASSIVE state. The third case of a collision-resolution interval is for multiple stations to request the channel causing a collision step: The stations in the allowed ID interval are once more split into two new ID intervals and the stack as well as the variables for each station are updated. In this case, the duration of the collision-resolution interval is equal to the collision time plus the channel delay.

The algorithm repeats the above three types of collision-resolution steps, until all the RTS collisions have been resolved. As soon as the backoff stack becomes empty and there are no values in the allowed ID interval, all stations know that all the collisions have been resolved. Accordingly, once the tree-splitting algorithm terminates, all stations are either in the PASSIVE state, or in the BACKOFF state if they have packets to send but were not in the RTS state at the beginning of the first collision that started the tree-splitting algorithm. The next access to the channel is driven by the arrival of new packets to the stations and the transmission of RTSs that have been backed off.

To permit the transmission of packet bursts, CARMA enforces waiting periods on receiving stations at strategic points in the operation of the protocol. A station that has received a data packet in the clear must wait for one maximum propagation time after processing a data packet, this allows the sender to send more packets if desired. A station that has understood any control packet must wait for twice the duration of the maximum propagation time; this allows correct RTS/CTS exchanges to take place. On the other hand, if a transmitting station is in the RTS state, the protocol enforces a waiting period of two maximum propagation times after sending its RTS. This allows the destination to receive the RTS and transmit the corresponding CTS. A sending station must also wait one maximum propagation time after the last data packet of its packet train.

2.2. Example of CARMA's Operation

We illustrate CARMA using a simple example (see Fig. 2). Each station has a distinct position in the leaves of a binary tree based on its ID. If n is the total number of stations in the system, the binary tree has $2n+1$ nodes. The root of the tree is labeled as n_r and its right and left child as n_1 and n_0, respectively. For each of the other nodes, the labels are composed of the parent label, plus a 0 if it is the left child or a 1 if it is a right child. As an example, take a system with four stations labeled n_{00}, n_{01}, n_{10}, and n_{11}. The binary tree has a total of seven nodes with the four stations as its leaves. The root of the tree has the label n_r. The left child of the root node is n_1 while its right child is n_0. Station n_0 is the parent node of n_{01} and n_{00}. Similarly, station n_{10} is the right child of node n_1, while station n_{11} is its left child. We define the subtree T_{label} as the subtree at node n_{label}. In our example, the subtree for node n_{01} is T_{01}. The backoff stack is empty and the allowed ID interval contains all the stations in the network, i.e., the allowed ID interval is (n_{00}, n_{11}).

Assume that, at time t_o, we are at node n_r and we are allowed to listen simultaneously at all the stations of its subtree T_r for a time period of τ seconds. Only one of the following three cases can occur:

- Case 1–Idle: There are no RTSs in any of the leaves (stations) in subtree T_r; therefore, the channel is idle. This lasts an idle transmission period T_i.
- Case 2–Success: There is only one RTS in the subtree T_r; therefore, there is no collision and a station acquires the floor. This lasts one successful transmission period T_s.
- Case 3–Collision: There are two or more stations (leaves) in the subtree T_r sending an RTS; therefore, a collision occurs. This lasts one failed transmission period T_f.

At time t_0, the first collision occurs (Step 1 in Fig. 2) with station n_{00} and n_{01} each sending an RTS, while station n_{10} and station n_{11} do not request the channel. All stations notice the beginning of the collision-resolution algorithm and update their stacks and their *LowID* as well as their *HiID* values. Stations n_{00} and n_{01} are members of the backoff ID interval; therefore, they both are on hold, because they must wait until the collisions in the allowed ID interval are resolved. They both are excluded from sending RTSs. After a time period T_f, Stations n_{10} and n_{11} are allowed to request the channel. Since stations n_{10} and n_{11} are in tree T_1 and do not wish the channel, an idle period occurs (Step 2 in Fig. 2). After $T_i = 2\tau$ seconds, all stations notice that the channel is idle, which means that there were no collisions; accordingly, the stations in the system must update their intervals and the stack. They execute a POP-stack command and the new allowed interval is (n_{00}, n_{01}); therefore, T_0 can proceed to solve its RTS collisions. Both stations n_{00} and n_{01} transmit an RTS control packet (Step 3 in Fig. 2) and another collision occurs. Because a collision occurred, the allowed

ID interval is split, i.e., the subtree T_0 is split into T_{00} and T_{01}. Station n_{01} is within the allowed interval while the n_{00} station must wait, its interval is the top of the stack. Since T_{01} has only one station requesting the channel, that station acquires the floor and transmits its data package (Step 4 in Fig. 2). After T_s seconds the stations in CARMA do an update, a POP-stack command. Station n_{00} can request and acquire the channel transmitting its data packet (Step 5 in Fig. 2). The termination of the collision-resolution phase is determined by an empty stack and an empty allowed ID interval. The stations empty their stacks and update the allowed ID interval permitting all stations to contend in the next round of contention. The transmission periods for each of the stations and the channel are illustrated in Fig. 2.

2.3. Handling Hidden Terminals

Our description of CARMA thus far has assumed that all stations can listen to one another. In this section we summarize a simple modification of the basic scheme intended for wireless LANs (WLANs) in which some stations may be hidden from others.

Handling hidden terminals in CARMA is more difficult than in other MAC protocols based on collision avoidance (e.g., MACAW [3], MACA [8], FAMA [5]) because all stations whose transmissions can impact one another must agree on the same state of the collision resolution algorithm, rather than just agreeing on allowing a given sender to transmit a packet without interference.

Fullmer and Garcia-Luna-Aceves [6] have demonstrated sufficient conditions for MAC protocols based on RTS/CTS exchanges to eliminate unwanted collisions due to hidden terminals. For the case of MAC protocols that use carrier sensing, a CTS must be longer than the length of an RTS, plus a maximum round-trip time and processing delays (including radio turn-around times). The net effect of the CTS length suggested in [6] is that of a single-channel, receiver-based busy tone that forces hidden sources to back off after they are forced to listen to at least part of the CTS sent by a receiver that has accepted the RTS from a given sender. Because the CTS can be only partially overheard by a station after it sends a failed RTS, it cannot be used to convey queue and channel state information.

Consider the case of a WLAN in which stations are trying to communicate with one another over a single hop, or with a base station to reach hidden nodes. The base station is in line of sight of every other station. In this case, each sender is in line of sight of the intended receiver (another station or the base station) and all stations should agree on the same state of a single transmission queue and the same state of the channel. For CARMA to work correctly in this case, the base station must send a second control packet, which we call the *collision resolution state* packet (CRS) immediately after it sends or hears a CTS, and immediately after detecting a collision of RTSs. The base station remains silent during idle collision-resolutions steps. The CRS contains the information on the state of the collision-resolution algorithm as perceived by the receiver. The CRS specifies if the collision-resolution step was successful or not; in the case of a successful collision-resolution step, the CRS indicates the identifier of the successful sender of the RTS.

3. Average Number of Collision Resolution Steps

In this section, we present the average number of steps (i.e., average number of collision steps, average number of idle steps, and average number of success steps) needed for a collision resolution algorithm based on a deterministic tree-splitting algorithm to resolve m collisions in a network of n stations, each having a unique ID. We also present upper bounds for the average number of steps needed for CARMA.

For the purpose of our analysis, we assume that (a) every station can listen to every other station, (b) the channel introduces no errors, so packet collisions are the only source of errors, (c) stations detect such collisions perfectly, (d) two or more transmissions that overlap in time in the channel must all be re-transmitted, (e) a packet propagates to all stations in exactly τ seconds [11], and (f) no failures occur.

The binary tree is a structure defined on a finite set of nodes composed of three disjoint sets: a root node, a binary tree called a left subtree and a binary tree called a right subtree. As we have described, there are only three possible cases to consider for the resolution of collisions: idle, success, or collision. For each of these cases, we wish to find a recursive equation expressing the average number of steps needed during the collision-resolution phase.

Consider a system with n stations and m RTSs arriving during a contention time period. Because each station in the system is assigned one or no RTS at any given time, a leaf of the binary tree, is assigned an "RTS" or an "idle," depending on whether or not it has an RTS to send. $\overline{\mathcal{Z}}(n,m)$ denotes the average number of idle steps, $\overline{\mathcal{C}}(n,m)$ denotes the average number of collision steps, and $\overline{\mathcal{S}}(n,m) = m$ is the number of success steps of the collision-resolution algorithm needed to resolve m collisions in a network of n stations. These three values depend on the number of leaves n and the number of stations with RTSs, m. They represent an average number of steps over all the possible permutations of m RTSs in n total stations.

The number of permutations of a tree with four leaves ($n = 4$), given that two out of the four stations are requesting the channel simultaneously ($m = 2$), is six. In our prior example (see Fig. 2), stations n_{00} and n_{01} each sends an RTS in the same time slot, while station n_{10} and station n_{11} remain idle. This tree represents just one of the six possible permutations. Let us assign the index i to the ith permutation and calculate $\mathcal{S}_i(4,2)$, $\mathcal{Z}_i(4,2)$ and $\mathcal{C}_i(4,2)$, respectively. Starting with the root node, the first thing that happens is a collision because two stations are competing for the channel. Following the rules of the algorithm, we take subtree T_1 and none of the stations in the subtree request the channel. We proceed with subtree T_0 and once again a collision occurs. The total number of collision steps for CARMA is $\mathcal{C}_i(4,2) = 2$. The next step is to go to node n_{00} and transmit its data packet. CARMA still has one more step to go: Node n_{01} transmits a data packet and CARMA terminates the collision-resolution phase. Therefore, for the example, the number of success steps is $\mathcal{S}_i(4,2) = 2$, the total number collision steps is $\mathcal{C}_i(4,2) = 2$, and the total number of idle steps is $\mathcal{Z}_i(4,2) = 1$.

In the above example, we have counted the number of steps that have collisions $(\mathcal{C}_i(4,2))$, the number of steps that only have one RTS in them $(\mathcal{S}_i(4,2))$ as well as the number of steps that are idle $(\mathcal{Z}_i(4,2))$ for the ith permutation. The same counting procedure can be repeated for each of the $\binom{4}{2} = 6$ permutations of trees with $n = 4$ and $m = 2$. The six possible combinations contribute equally to the total average number of steps $\overline{\mathcal{C}}(4,2)$, $\overline{\mathcal{S}}(4,2)$ and $\overline{\mathcal{Z}}(4,2)$. In general, the average for each of the three types of steps can be calculated by adding each individual permutation cost and dividing by the total number of permutations.

For counting purposes, a subtree that has no RTS stations or only one RTS station does not have to be explored further. Counting can stop there and one unit can be added to either $\mathcal{Z}$ or $\mathcal{S}$. $\mathcal{S}$ is always equal to m for CARMA. Based on this example, we can deduce the general rules shown in Table 1.

3.1. Average Number of Success Steps

Any tree or subtree, regardless of its size, has a success cost of one step if there exist only one station with an RTS to send. This is the case because we need to visit the root node of such a tree or subtree and stop there. In the case of a tree with m stations with an RTS to send, there are exactly m subtrees with one RTS each; otherwise, they would be a subtree with no RTS nodes or more than two RTSs creating a collision. For both of this cases the number of success steps is $\mathcal{S} = 0$. Table 2 shows the number of success steps for a system with $n = 4$ and $m = 2$. As Table 2 shows, the number of success steps is always 2. The total average number of success steps for any tree of size n with m RTS stations is simply Rule 5 from Table 1, i.e., $\overline{\mathcal{S}}(n,m) = m$.

3.2. Average Number of Collision Steps

In our example of CARMA's operation, we found the number of collision steps to be $\mathcal{C}_i(4,2) = 2$. This tree can be visualized as two disjoint binary subtrees and the parent node. Similarly, the total number of collision steps can be expressed as the number of collision steps of the right subtree, plus the number of collision steps of the left subtree plus one step for the root of the tree. This result can be extended to the general case, yielding the following equation:

$$\mathcal{C}_{root\ tree} = \mathcal{C}_{right\ subtree} + \mathcal{C}_{left\ subtree} + 1 \tag{3.1}$$

In addition to our example, there are five other possible ways to distribute two stations with an RTS to send in four positions. Table 2 shows all six cases and the number of collision steps $\mathcal{C}_i$ associated with each of them.

To calculate $\overline{\mathcal{C}}(4,2)$, we sum each individual permutation and divide this result by six. Notice that we have four cases in which the right and left subtrees each has one RTS station. There is one case in which the left subtree has two RTS stations and one case in which the right subtree has two RTS stations. The average number of collision steps $\overline{\mathcal{C}}(4,2)$ can be expressed as follows:

$$\overline{\mathcal{C}}(4,2) = \sum_{i=0}^{2} \frac{\binom{2}{2-i}\binom{2}{i}}{\binom{4}{2}} \left[\overline{\mathcal{C}}\,(2, 2-i) + \overline{\mathcal{C}}\,(2, i) + 1\right] \tag{3.2}$$

The recursion splits the original tree for $n = 4$ into two subtrees, each for $n = 2$, plus the root node. $\frac{\binom{2}{2-i}\binom{2}{i}}{\binom{4}{2}}$ is the probability of having $2-i$ RTS stations in one subtree and i RTS stations on the other subtree. The following Theorem extends this result to the average number of collision steps $\overline{\mathcal{C}}(n,m)$; its proof is in the Appendix.

Theorem 3.1. For all $n \geq m > 1$, where n is the number of stations and m is the number of stations participating in the tree splitting algorithm, the average number of collision steps required to resolve all m collisions is

$$\overline{\mathcal{C}}(n,m) = \sum_{i=\mu}^{\nu} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\overline{\mathcal{C}}\,(\alpha, m-i) + \overline{\mathcal{C}}\,(\beta, i) + 1\right] \tag{3.3}$$

where

$$\alpha = \lceil n/2 \rceil; \quad \beta = n - \alpha = n - \lceil n/2 \rceil$$
$$\mu = \begin{cases} 0 & \text{if } m \leq \alpha \\ m - \alpha & \text{if } m > \alpha \end{cases}$$
$$\nu = \begin{cases} m & \text{if } m \leq \beta \\ \beta & \text{if } m > \beta \end{cases}$$

The proof of this theorem is in the Appendix.

3.3. Average Number of Idle Steps

According to Rule 8 in Table 1, if there is only one RTS leaf ($m = 1$), the number of idle steps is zero, regardless of the size of tree or subtree. In our example of the binary tree with four leaves ($n = 4$) and two stations with an

RTS to send ($m = 2$), we can plot a cost table with all the six permutation cases, as shown in Table 2. The number of idle steps at the root node can be expressed as the number of idle steps for the right subtree, plus the number of idle steps for the left subtree. This result can be extended to the general case, yielding the following equation:

$$\mathcal{Z}_{root\ tree} = \mathcal{Z}_{right\ subtree} + \mathcal{Z}_{left\ subtree} \tag{3.4}$$

To calculate the average number of idle steps for our example, we sum each of the individual number of idle steps for each permutation and divide this result by six permutations. There are four cases in which the right and left subtrees each has one station with an RTS to send. There is one case in which the left subtree has two stations with an RTS to send, and one case in which the right subtree has two stations with an RTS to send. The average number of idle steps $\overline{\mathcal{Z}}(4,2)$ can be expressed by

$$\overline{\mathcal{Z}}(4,2) = \sum_{i=0}^{2} \frac{\binom{2}{2-i}\binom{2}{i}}{\binom{4}{2}} \left[\overline{\mathcal{Z}}\left(2, 2-i\right) + \overline{\mathcal{Z}}\left(2, i\right)\right] \tag{3.5}$$

Theorem 3.2 below extends the above to the average number of idle steps $\overline{\mathcal{Z}}(n,m)$. The proof of this theorem is in the Appendix.

Theorem 3.2. For all $n \geq m > 1$, where n is the number of stations and m is the number of stations participating in the tree splitting algorithm, the average number of idle steps required to resolve all m collisions is

$$\overline{\mathcal{Z}}(n,m) = \sum_{i=\mu}^{\nu} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\mathcal{Z}\left(\alpha, m-i\right) + \mathcal{Z}\left(\beta, i\right)\right] \tag{3.6}$$

There exist a relationship between each of the three types of steps that can be summarized as

$$\mathcal{S}(n,m) + \mathcal{Z}(n,m) - \mathcal{C}(n,m) - 1 = 0 \tag{3.7}$$

Notice that this equation not only applies to the average number of steps, but also to any specific ith permutation.

3.4. Upper Bounds

Obviously, the upper bound for the average number of success steps $\overline{\mathcal{S}}(n,m)$ is equal to m for CARMA. The following two theorems provide upper bounds on the average number of idle and collision steps needed, which are independent of the number of stations in the network. The Appendix contains the proof of these bounds, which improve on the bounds previously reported by Garces and Garcia-Luna-Aceves [7].

Theorem 3.3. For all $n \geq m > 1$, where n is the number of stations and m is the number of stations participating in the tree splitting algorithm, the average number of idle steps in CARMA is bounded by $\overline{\mathcal{Z}}(n,m) \leq 0.443m$.

Theorem 3.4. For all $n \geq m > 1$, where n is the number of stations and m is the number of stations participating in the tree splitting algorithm, the average number of collision steps in CARMA is bounded by $\overline{\mathcal{C}}(n,m) \leq 1.443m - 1$.

It follows from these two theorems and the fact that m successes are needed to resolve m requests for the transmission queue that the total number of collision-resolution steps needed to resolve m RTS collisions is smaller than $2.886m - 1$ steps. This clearly indicates that the type of collision resolution used in CARMA is very efficient. Furthermore, these bounds are independent of the number of stations in the network, which we use to provide an approximate throughput analysis.

4. Approximate Throughput

The analysis in this section makes the same assumptions introduced in the previous section and uses the same traffic model used for the FAMA-NTR protocol [5]. Given that the upper bounds on the average number of steps (success steps, collision steps, and idle steps) in a collision-resolution period are independent of the number of stations, we approximate the traffic into the channel, with an infinite number of stations, each having at most one RTS to send at any time, and forming a Poisson source sending RTSs with an aggregate mean generation rate of λ packets per unit time. With this model, the average number of RTS arrivals in a time interval of length T is λT, i.e., $m = \lambda T$. All data blocks have a duration of δ seconds. The RTS and CTS packets are the same size with a duration of γ seconds. The average channel throughput is given by

$$S = \frac{\overline{U}}{\overline{B} + \overline{I}} \tag{4.1}$$

where $\overline{U}$ is the average utilization time of the channel, during which the channel is being used to transmit data packets; $\overline{B}$ is the expected duration of a busy period, during which the channel is busy with successful or tree-transmission periods; and $\overline{I}$ is the average idle period, i.e., the average interval between two consecutive busy periods.

4.1. Unslotted CARMA

Theorem 4.1. The throughput of unslotted CARMA is bounded from below by

$$S \geq \frac{\delta\lambda e^{-\lambda\tau}(\lambda\tau - 1) - \delta\lambda^2\tau}{A \cdot e^{-\lambda\tau} + B} \tag{4.2}$$

where

$$\begin{aligned} A &= -\lambda\delta - 3\lambda\gamma - 5\lambda\tau + \lambda^2\tau\delta + 3.433\gamma\lambda^2\tau + 6.732\tau^2\lambda^2 \\ B &= -6.732\lambda^2\tau^2 - 3.433\gamma\lambda^2\tau - 1 - \lambda^2\tau\delta + \gamma\lambda \end{aligned}$$

Proof. A successful transmission consists of an RTS with one propagation delay to the intended recipient, a CTS with a propagation delay to the sender, and a data packet followed by a propagation delay. Therefore, the average duration period for a successful transmission is $T_s = 2\gamma + 3\tau + \delta$.

For an RTS to be successful, it must be the only packet in the channel during its transmission. Its probability of success equals the probability that no arrivals occur in τ seconds, because there is a delay across the channel of τ seconds before all the other stations in the network detect the carrier signal. After this vulnerability period of τ seconds, all stations defer their transmissions. Therefore, given that arrivals of RTSs to the channel are Poisson with parameter λ, we obtain the probability of success $P_s = e^{-\lambda\tau}$.

The number of stations that participate in the collision-resolution phase are $m = \lambda\tau$. Within the tree, the three cases of the collision resolution discussed in the previous sections are present. Each one of them has an average upper bound on the number of steps that is independent of the number of stations (n), but is a function of the number of stations requesting the channel (m). In the case of a colliding transmission ($m > 1$), the time period consists of one RTS package followed by one or more RTSs transmitted by other stations within time Y, where $0 \leq Y \leq \tau$, plus one propagation delay τ. The average failed transmission period is bounded by [5]; $T_f \leq \gamma + 2\tau$

A waiting period of 2τ seconds is required after a successful transmission or a tree transmission. A busy period is composed of both the successful and the tree transmission periods. The duration of an average busy period equals the sum of the percentage of successful transmission periods times their duration, T_s, plus the percentage of the tree periods times their duration. The tree periods are composed of three parts, corresponding to success, idle, and collision periods, each with a distinct cost and duration. According to the upper bounds derived in section 3.4 for CARMA, the average busy period can be bounded as follows:

$$\overline{B} \leq T_s \cdot P_s + \left(T_s m + T_f(1.433 * m - 1) + T_i 0.433 * m\right)(1 - P_s) \tag{4.3}$$

Substituting the values for P_s, T_f, T_s, T_i and m, we obtain

$$\overline{B} \leq \left(\delta + 3\gamma + 5\tau - \lambda\tau\delta - 3.433\lambda\tau\gamma - 6.732\tau^2\lambda\right) \cdot e^{-\lambda\tau} + \left(\lambda\tau\delta + 3.433\lambda\tau\gamma + 6.732\tau^2\lambda - \gamma - 2\tau\right) \quad (4.4)$$

The channel carries user data for δ seconds each time an RTS is sent successfully without collision resolution, and δ seconds for each of the RTSs that collide when collision resolution is applied. Therefore, the average utilization is:

$$\overline{U} = \delta \cdot P_s + \delta m(1 - P_s) = (1 - \lambda\tau)\delta \cdot e^{-\lambda\tau} + \delta\lambda\tau \quad (4.5)$$

The average idle period is equal to the average inter-arrival time plus the average waiting period enforced, therefore, $\overline{I} \leq \frac{1}{\lambda} + 2\tau$.

Substituting Eqs. (4.3), (4.5) and $\overline{I}$ into Eq. (4.1) a lower bound of average throughput of CARMA is given by Eq. (4.2). □

4.2. Slotted CARMA

Theorem 4.2. The throughput of slotted CARMA is bounded from below by

$$S \geq \frac{\delta\lambda^2\tau^2 e^{-\lambda\tau} - \delta\lambda\tau}{A \cdot e^{-\lambda\tau} + B} \quad (4.6)$$

where

$$A = \tau + 0.433\lambda\tau\gamma + 1.299\lambda\tau^2 + \lambda^2\tau^2\delta + 3.433\lambda^2\tau^2\gamma + 5.299\lambda^2\tau^3 - \gamma$$
$$B = -2.0\tau - \lambda\tau\delta - 3.433\lambda\tau\gamma - 5.299\lambda\tau^2 + \gamma$$

Proof. The same assumptions used for unslotted CARMA can be used for slotted CARMA. The channel is slotted and each slot lasts as long as the maximum propagation delay τ. With slotting, stations are restricted to start transmissions only on slot boundaries.

As it was the case in unslotted CARMA, the average duration period for a successful transmission consists of an RTS with one propagation delay to the intended recipient, a CTS with a propagation delay to the sender, and a data packet followed by a propagation delay. Therefore, the average duration period for a successful transmission is $T_s = \delta + 2\gamma + 3\tau$. The probability that an RTS is successful is

$$P_s = P\{k = 1 \text{ arrival in a slot} | \text{some arrivals in a slot}\} = \frac{\lambda\tau \cdot e^{-\lambda\tau}}{1 - e^{-\lambda\tau}} \quad (4.7)$$

In the case of a colliding transmission ($m > 1$), the time period consists of one RTS followed by a propagation delay τ. All colliding RTSs are sent at the beginning of the same slot; accordingly, we have

$$T_f = \gamma + \tau \tag{4.8}$$

As it was done for unslotted CARMA, $\overline{B}$ can be bounded according to Eq. (4.3). Substituting the values for P_s, T_f, T_s, T_i and m, we obtain

$$\begin{aligned}\overline{B} &\leq T_s \cdot P_s + (T_s m + T_f(1.433m - 1) + T_i 0.433m)\,(1 - P_s)\\ &= \frac{(-0.433\lambda\tau\gamma - 1.299\lambda\tau^2 - \lambda^2\tau^2\delta - 3.433\lambda^2\tau^2\gamma - 5.299\lambda^2\tau^3 + \gamma + \tau)e^{-\lambda\tau}}{(1 - e^{-\lambda\tau})} + \\ &\quad \frac{(\lambda\tau\delta + 3.433\lambda\tau\gamma + 5.299\lambda\tau^2 - \gamma - \tau)}{1 - e^{-\lambda\tau}}\end{aligned} \tag{4.9}$$

The average utilization is:

$$\overline{U} = \delta \cdot P_s + \delta m(1 - P_s) = \frac{\delta\lambda\tau(1 - \lambda\tau \cdot e^{-\lambda\tau})}{(1 - e^{-\lambda\tau})} \tag{4.10}$$

The average idle period is the same as in the slotted FAMA-NTR protocol [5] with the modification that we require a waiting period of 2τ after each collision-resolution period, i.e.,

$$\overline{I} \leq \tau \cdot \frac{1}{1 - e^{-\lambda\tau}} + 2\tau \tag{4.11}$$

Substituting Eqs. (4.9), (4.10),and (4.11) into Eq. (4.1) a lower bound on the average throughput of slotted CARMA can be written as in Eq. (4.6). $\square$

5. Numerical Results

We compare CARMA with FAMA-NTR for the cases of a low-speed network (9600 b/s) and high-speed network (1Mb/s) in which either small data packets (53 bytes) or large data packets (400 bytes) are transmitted. We assume the distance between stations to be the same and define the diameter of the network to be 1 mile. Assuming these parameters, the propagation delay of the channel is 5.4μs. In order to accommodate the use of IP addresses for destination and source, the minimum size of an RTSs and a CTSs is 20 bytes. We normalize the throughput results by defining the following variables

$$\begin{aligned} a &= \frac{\delta}{\tau} && \text{(normalized data packets)} \\ b &= \frac{\gamma}{\tau} && \text{(normalized control packets)} \\ G &= \lambda\tau && \text{(normalized offered load)} \end{aligned} \tag{5.1}$$

Substituting the new normalized variables from Eq. (5.1) into Eqs. (4.2) and (4.6), respectively, we list in Table 3 the normalized throughput for each protocol for both the slotted and the unslotted versions. Our results on FAMA throughput are in agreement with the results presented in [5] with the exception of the $\overline{I}$ and T_f which are bounded as in CARMA. Table 4 summarizes the protocol parameters used in our comparison.

Figs. 3 and 4 show the average throughput (S) versus the offered load (G) for CARMA and FAMA-NTR. It is clear that slotting does not provide much performance improvement compare to unslotted in CARMA. To achieve high throughput, the size of the control packets need to be small compared to the length of the data packets or packet trains. CARMA behaves like FAMA-NTR when the offered load is small. As the offered load increases, the throughput of FAMA-NTR decreases rapidly, while CARMA initially decreases reaching a constant throughput value. This value is obtained by taking $\lim_{\lambda\to\infty} S$. For unslotted CARMA we have

$$\lim_{\lambda\to\infty} S = \lim_{G\to\infty} S = \frac{a}{a + 3.433b + 6.732} = \frac{\delta}{\delta + 3.433\gamma + 6.732\tau} \tag{5.2}$$

while for the slotted CARMA

$$\lim_{\lambda\to\infty} S = \lim_{G\to\infty} S = \frac{a}{a + 3.433b + 5.295} = \frac{\delta}{\delta + 3.433\gamma + 5.295\tau} \tag{5.3}$$

For slotted as well as for unslotted FAMA-NTR, $\lim_{\lambda\to\infty} S_{fama} = 0$. If we were to have a perfect FAMA protocol in which no collisions of RTSs ever occurred, and a constant flow of data packets, the best possible throughput would be

$$S_{max} = \frac{\delta}{\delta + 2\gamma + 3\tau} \tag{5.4}$$

The ratio of $\lim_{\lambda\to\infty} S$ to S_{max} for unslotted CARMA is then

$$1 \geq \frac{S}{S_{max}} \geq \frac{\delta + 2\gamma + 3\tau}{\delta + 3.433\gamma + 6.732\tau} \tag{5.5}$$

The above result is very encouraging since $\tau \ll \gamma \ll \delta$. The result also indicates an improvement if the parameter b is small compared to a. In practice, this effect can be achieved by allowing a station to transmit longer data packets or multiple packets per floor acquisition.

To verify that the throughput values of CARMA approximated using an infinite population and the upper bounds on average number of steps required for collision resolution times provides a good lower bound for any traffic load, we simulated slotted CARMA using 100 stations that generate RTSs according to a Poisson probability distribution function. The simulations were done ten times for each given $m = \tau\lambda$ value to insure convergence. The results of the simulation are shown in Fig. 4 only for the case of long data packets in a high-speed network, and indicate that our analysis provides a very good approximation of the average throughput.

6. Conclusions

CARMA implements a three-way handshake based on small control packets between sender and receiver, plus a deterministic collision resolution algorithm ensuring that there is always a successful RTS during each busy period. Our simulation validates the simplifying assumptions made to obtain a lower bound of average throughput as a function of channel load. Our analysis shows that collision resolution improves the performance of FAMA protocols substantially. The reason is that the average time it takes for a collision-resolution algorithm to resolve the collisions of RTSs is much smaller than the time used to transmit the associated data packet trains, which are sent with no collisions due to floor acquisition. We have shown that, as the arrival rate of RTSs increases, the throughput achieved by CARMA is close to the maximum throughput that any FAMA protocol can achieve if propagation delays and the control packets used to acquire the floor are much smaller than the data packet trains sent by stations. Our analysis of the average number of collisions and idle steps needed to resolve m collisions in a net of n nodes extends prior work on collision resolution, which focused in the total average number of steps needed [4]. Obtaining these results is essential to the accurate analysis of MAC protocols in which control packets used to resolve collisions have different length than data packets.

Appendix

Proof of Theorem 3.1. It is trivial to show that $\overline{\mathcal{C}}(n,0)$ and $\overline{\mathcal{C}}(n,1)$ equal 0 for all $n \geq 1$. If we have two stations and both are RTS stations the average number of collision steps is $\overline{\mathcal{C}}(2,2) = 1$. With this initial conditions and following the tree-splitting algorithm we are in a position to find the average number of collision steps for $\overline{\mathcal{C}}(3,2)$. Since $m = 2$, we have to split $n = 3$ into two subtrees of size $\alpha = 2$ and $\beta = 1$, respectively. Therefore, we can either have two RTS stations in the α-split and none in the β-split; or one RTS station in the α-split and one in the β-split. The probability of each of these cases is $P\{(2 \in \alpha\text{-split}) \wedge (0 \in \beta\text{-split})\} = \frac{\binom{2}{2}\binom{1}{0}}{\binom{3}{2}}$ and $P\{(1 \in \alpha\text{-split}) \wedge (1 \in \beta\text{-split})\} = \frac{\binom{2}{1}\binom{1}{1}}{\binom{3}{2}}$, respectively. For each of these cases, the probability of the split must be multiply by the average number of collision steps of the right subtree plus the average number of collision steps for the left subtree plus one collision step for the root of both subtrees. Therefore,

$$\begin{aligned}
\overline{\mathcal{C}}(3,2) &= P\{(2 \in \alpha\text{-split}) \wedge (0 \in \beta\text{-split})\}\left[\overline{\mathcal{C}}(2,2) + \overline{\mathcal{C}}(1,0) + 1\right] + \\
&\quad P\{(1 \in \alpha\text{-split}) \wedge (1 \in \beta\text{-split})\}\left[\overline{\mathcal{C}}(2,1) + \overline{\mathcal{C}}(1,1) + 1\right] \\
&= \frac{\binom{2}{2}\binom{1}{0}}{\binom{3}{2}}\left[\overline{\mathcal{C}}(2,2) + \overline{\mathcal{C}}(1,0) + 1\right] + \frac{\binom{2}{1}\binom{1}{1}}{\binom{3}{2}}\left[\overline{\mathcal{C}}(2,1) + \overline{\mathcal{C}}(1,1) + 1\right] \\
&= \sum_{i=0}^{1} \frac{\binom{2}{2-i}\binom{1}{i}}{\binom{3}{2}}\left[\overline{\mathcal{C}}(2,2-i) + \overline{\mathcal{C}}(1,i) + 1\right] \qquad (6.1)
\end{aligned}$$

Now we assume that Eq. (3.3) is satisfied for all $\beta \le \alpha \le \lceil \frac{n}{2} \rceil$ and show that Eq. (3.3) holds for n. If n is even, $\alpha = \beta = \frac{n}{2}$; otherwise, $\beta = \alpha - 1$. The probability that $m - i$ RTS stations are in the α-split while the remaining i RTS stations are in the β-split is given by

$$P\{(m-i \in \alpha\text{-split}) \wedge (i \in \beta\text{-split})\} = \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \qquad (6.2)$$

Therefore, the average number of collision steps for this specific split (i.e., $m - i$ RTS stations in the α-split and i RTS stations in the β-split) equals the average number of collision steps for the α-split, plus the average number of collision steps for the β-split, plus one step for the root of both subtrees, times the probability of such a split, or

$$\frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}}\left[\overline{\mathcal{C}}(\alpha, m-i) + \overline{\mathcal{C}}(\beta, i) + 1\right] \qquad (6.3)$$

For the average number of collision steps, $\mathcal{C}(n,m)$, we need to consider the number of collision steps as well as the probability of each of the possible α- and β-splits; therefore,

$$\begin{aligned}\overline{\mathcal{C}}(n,m) = &\frac{\binom{\alpha}{m-\mu}\binom{\beta}{\mu}}{\binom{n}{m}}\Big[\overline{\mathcal{C}}\,(\alpha, m-\mu) + \overline{\mathcal{C}}\,(\beta,\mu) + 1\Big] + \\ &\frac{\binom{\alpha}{m-\mu-1}\binom{\beta}{\mu+1}}{\binom{n}{m}}\Big[\overline{\mathcal{C}}\,(\alpha, m-\mu-1)) + \overline{\mathcal{C}}\,(\beta,\mu+1) + 1\Big] + \ldots\ldots + \\ &\frac{\binom{\alpha}{m-\nu+1}\binom{\beta}{\nu-1}}{\binom{n}{m}}\Big[\overline{\mathcal{C}}\,(\alpha, m-\nu+1) + \overline{\mathcal{C}}\,(\beta,\nu-1) + 1\Big] + \\ &\frac{\binom{\alpha}{m-\nu}\binom{\beta}{\nu}}{\binom{n}{m}}\Big[\overline{\mathcal{C}}\,(\alpha, m-\nu) + \overline{\mathcal{C}}\,(\beta,\nu) + 1\Big] \end{aligned} \tag{6.4}$$

There are three possible μ-ν combinations. First, if $m \le \alpha$ and $m \le \beta$, then $\mu = 0$ and $\nu = m$. In the second case, $m \le \alpha$ and $\beta < m$; therefore, $\mu = 0$, while $\nu = \beta$. Finally, if m is greater than both α and β, then $\mu = m - \alpha$ and $\nu = \beta$. Note that the parameter m cannot be $> \alpha$ and $\le \beta$ at the same time because $\beta \le \alpha$; accordingly, this case is excluded. The sum of the average number of steps for each of the possible splits yields Eq. (3.3). □

Proof of Theorem 3.2. For all $n \ge 1$, $\overline{\mathcal{Z}}(n,0) = 1$ and $\overline{\mathcal{Z}}(n,1) = 0$. If we have two stations and both of the stations wish to request the channel, the average number is $\overline{\mathcal{Z}}(2,2) = 0$, because resolving this collisions requires one collision step and two success steps. With these initial conditions, we can find the average number of idle steps for $\overline{\mathcal{Z}}(3,2)$. Because $m = 2$, we have to split $n = 3$ into two sub-trees of size $\alpha = 2$ and $\beta = 1$, respectively. Therefore, we can either have two RTS stations in the α-split and none in the β-split; or one RTS station in the α-split and one in the β-split. The probability of each of these cases is $P\{(2 \in \alpha\text{-split}) \wedge (0 \in \beta\text{-split})\} = \frac{\binom{2}{2}\binom{1}{0}}{\binom{3}{2}}$ and $P\{(1 \in \alpha\text{-split}) \wedge (1 \in \beta\text{-split})\} = \frac{\binom{2}{1}\binom{1}{1}}{\binom{3}{2}}$, respectively. Therefore,

$$\begin{aligned}\overline{\mathcal{Z}}(3,2) &= P\{(2 \in \alpha\text{-split}) \wedge (0 \in \beta\text{-split})\}\Big[\overline{\mathcal{Z}}(2,2) + \overline{\mathcal{Z}}(1,0) + 1\Big] + \\ &\quad P\{(1 \in \alpha\text{-split}) \wedge (1 \in \beta\text{-split})\}\Big[\overline{\mathcal{Z}}(2,1) + \overline{\mathcal{Z}}(1,1) + 1\Big] \\ &= \frac{\binom{2}{2}\binom{1}{0}}{\binom{3}{2}}\Big[\overline{\mathcal{Z}}\,(2,2) + \overline{\mathcal{Z}}\,(1,0)\Big] + \frac{\binom{2}{1}\binom{1}{1}}{\binom{3}{2}}\Big[\overline{\mathcal{Z}}\,(2,1) + \overline{\mathcal{Z}}\,(1,1) + 1\Big] \\ &= \sum_{i=0}^{1}\frac{\binom{2}{2-i}\binom{1}{i}}{\binom{3}{2}}\Big[\overline{\mathcal{Z}}\,(2,2-i) + \overline{\mathcal{Z}}\,(1,i)\Big] \end{aligned} \tag{6.5}$$

Now we assume that Eq. (3.6) is satisfied for all $\beta \le \alpha\lceil \frac{n}{2} \rceil$ and show that Eq. (3.6) holds for n. If n is even, $\alpha = \beta = \frac{n}{2}$; otherwise, $\beta = \alpha - 1$. The probability that $m - i$ RTS stations are in the α-split while the remaining i RTS stations are in the β-split is given by $P\left\{(m - i \in \alpha\text{-split}) \wedge (i \in \beta\text{-split})\right\} = \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}}$.

Therefore, the average number of idle steps for this specific split (i.e., $m-i$ RTS stations in the α-split and i RTS stations in the β-split) is equal to the average number of idle steps for the α-split, plus the average number of idle steps for the β-split, times the probability of such a split, or

$$\frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}}\left[\overline{\mathcal{Z}}\left(\alpha, m-i\right) + \overline{\mathcal{Z}}\left(\beta, i\right)\right] \tag{6.6}$$

For the average number of idle steps, $\mathcal{Z}(n, m)$, we need to consider the cost as well as the probability of each of the possible α- and β-splits; therefore,

$$\begin{aligned}\overline{\mathcal{Z}}(n,m) = {} & \frac{\binom{\alpha}{m-\mu}\binom{\beta}{\mu}}{\binom{n}{m}}\left[\overline{\mathcal{Z}}\left(\alpha, m-\mu\right) + \overline{\mathcal{Z}}\left(\beta, \mu\right)\right] + \\ & \frac{\binom{\alpha}{m-\mu-1}\binom{\beta}{\mu+1}}{\binom{n}{m}}\left[\overline{\mathcal{Z}}\left(\alpha, m-\mu-1)\right) + \overline{\mathcal{Z}}\left(\beta, \mu+1\right)\right] + \ldots.. + \\ & \frac{\binom{\alpha}{m-\nu+1}\binom{\beta}{\nu-1}}{\binom{n}{m}}\left[\overline{\mathcal{Z}}\left(\alpha, m-\nu+1\right) + \overline{\mathcal{Z}}\left(\beta, \nu-1\right)\right] + \\ & \frac{\binom{\alpha}{m-\nu}\binom{\beta}{\nu}}{\binom{n}{m}}\left[\overline{\mathcal{Z}}\left(\alpha, m-\nu\right) + \overline{\mathcal{Z}}\left(\beta, \nu\right)\right]\end{aligned} \tag{6.7}$$

The sum of the average number of steps for each of the possible splits yields Eq. (3.6). The parameters α, β, μ and ν are the same as in Eq. (3.3). $\square$

Proof of Theorem 3.3. The initial values for $\overline{\mathcal{Z}}$ and $\overline{\mathcal{C}}$ up to $n = 4$ are given in Table 5, each satisfying Theorems 3.1 and 3.2. Regardless of the value of n, $\overline{\mathcal{Z}}(n, 0) = 1$, $\overline{\mathcal{Z}}(n, 1) = 0$ and $\overline{\mathcal{Z}}(n, n) = 0$.

Now we assume that, for all $4 \leq n \leq \alpha$ and all $2 \leq m \leq \nu$, the conditions $\overline{\mathcal{Z}}(\alpha, m) \leq 0.443m$ are satisfied, and we show that the condition holds for all $\overline{\mathcal{Z}}(n, m)$. There are three cases to consider in Eq. (3.6) based on the summation indices.

Case 1: $m \leq \alpha$ and $m \leq \beta$: Then $\mu = 0$ while $\nu = m$. Therefore,

$$\frac{\overline{\mathcal{Z}}(n,m)}{m} = \sum_{i=0}^{m} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}}\left[\mathcal{Z}\left(\alpha, m-i\right) + \mathcal{Z}\left(\beta, i\right)\right] \tag{6.8}$$

Extracting the first two and the last two elements from the summation (i.e., the elements with $i = 0, 1, m-1, m$) and noting that $\overline{\mathcal{Z}}(\alpha, 1) = \overline{\mathcal{Z}}(\beta, 1) = 0$, and $\overline{\mathcal{Z}}(\alpha, 0) = \overline{\mathcal{Z}}(\beta, 0) = 1$, we obtain

$$\overline{\mathcal{Z}}(n,m) = \sum_{i=2}^{m-2} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\mathcal{Z}(\alpha, m-i) + \mathcal{Z}(\beta, i)\right] + \frac{\binom{\alpha}{m}\binom{\beta}{0}}{\binom{n}{m}} \left[\mathcal{Z}(\alpha, m) + 1\right] + \frac{\binom{\alpha}{m-1}\binom{\beta}{1}}{\binom{n}{m}} \mathcal{Z}(\alpha, m-1) + \frac{\binom{\alpha}{1}\binom{\beta}{m-1}}{\binom{n}{m}} \mathcal{Z}(\beta, m-1) + \frac{\binom{\alpha}{0}\binom{\beta}{m}}{\binom{n}{m}} \left[1 + \mathcal{Z}(\beta, m)\right] \tag{6.9}$$

Because $\overline{\mathcal{Z}}(\alpha, m) \leq 0.443m$ and $\overline{\mathcal{Z}}(\beta, m) \leq 0.443m$, we obtain

$$\overline{\mathcal{Z}}(n,m) \leq 0.443m \sum_{i=0}^{m} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} + \frac{\binom{\alpha}{m}\binom{\beta}{0}}{\binom{n}{m}} - 0.443\frac{\binom{\alpha}{m-1}\binom{\beta}{1}}{\binom{n}{m}} - 0.443\frac{\binom{\alpha}{1}\binom{\beta}{m-1}}{\binom{n}{m}} + \frac{\binom{\alpha}{0}\binom{\beta}{m}}{\binom{n}{m}} \tag{6.10}$$

For any binomial coefficient, $\binom{n}{k} = \frac{n-k+1}{k}\binom{n}{k-1}$. Therefore, noticing that the sum from Eq. (6.10) equals one, we have

$$\overline{\mathcal{Z}}(n,m) \leq 0.443m + \frac{\alpha - 0.443m\beta + 1 - m}{m} \frac{\binom{\alpha}{m-1}}{\binom{n}{m}} + \frac{\beta - 0.443m\alpha + 1 - m}{m} \frac{\binom{\beta}{m-1}}{\binom{n}{m}} \tag{6.11}$$

For the equation $\overline{\mathcal{Z}}(n,m) \leq 0.443m$ to be true, the last two terms in Eq. (6.11) must be zero or negative. If n is even, then $\alpha = \beta$ and

$$\overline{\mathcal{Z}}(n,m) \leq 0.443m + \frac{(2 - 0.886m)\alpha + (2 - 2m)}{m} \frac{\binom{\alpha}{m-1}}{\binom{n}{m}} \tag{6.12}$$

Because $m > 3$, $\alpha(2 - 0.886m) + (2 - 2m) < 0$; therefore, $\overline{\mathcal{Z}}(n,m) \leq 0.443m$. If n is odd, then $\beta = \alpha - 1$; accordingly, we have

$$\overline{\mathcal{Z}}(n,m) \leq 0.443m + \frac{\alpha(1 - 0.443m) + 1 - 0.557m}{m} \frac{\binom{\alpha}{m-1}}{\binom{n}{m}} + \frac{\alpha(1 - 0.443m) - m}{m} \frac{\binom{\beta}{m-1}}{\binom{n}{m}} \tag{6.13}$$

Because $m > 3$, $\alpha(1 - 0.443m) + 1 - 0.557m \leq 0$ and $\alpha(1 - 0.443m) - m \leq 0$; therefore, our assumption that $\overline{\mathcal{Z}}(n,m) \leq 0.443m$ is correct for any n and any $m > 1$.

Case 2: $m \leq \alpha$ and $m > \beta$: Then $\mu = 0$ while $\nu = \beta$ and n can only be odd. Further more, $\alpha = m$ and $\beta = m - 1$, while $n = 2m - 1$. Extracting the first term in the summation in Eq. (3.6) we obtain

$$\overline{\mathcal{Z}}(n,m) = \sum_{i=1}^{\beta} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\mathcal{Z}\left(\alpha, m-i\right) + \mathcal{Z}\left(\beta, i\right)\right] + \frac{\binom{\alpha}{m}\binom{\beta}{0}}{\binom{n}{m}} \tag{6.14}$$

where several terms have been eliminated since they equal zero. Because $\mathcal{Z}(m, m-i) \leq 0.443(m-i)$ and $\mathcal{Z}(m-1, i) \leq 0.443i$, we obtain

$$\overline{\mathcal{Z}}(n,m) \leq \sum_{i=1}^{\beta} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[0.443(m-i) + 0.443i\right] + \frac{\binom{\alpha}{m}\binom{\beta}{0}}{\binom{n}{m}} = 0.443m \tag{6.15}$$

We conclude that our assumption $\overline{\mathcal{Z}}(n,m) \leq 0.443m$ is correct for any n and all $m > 1$.

Case 3: $m > \alpha$ and $m > \beta$: Then $\mu = m - \alpha$ while $\nu = \beta$. Once again, we extract and evaluate the last member of the summation in Eq. (3.6) as well as set $\mathcal{Z}(\alpha, m-i) \leq 0.443(m-i)$ and $\mathcal{Z}(\beta, i) \leq 0.443i$. Therefore, we obtain

$$\overline{\mathcal{Z}}(n,m) = 0.443m - 0.443\beta \frac{\binom{\alpha}{m-\beta}\binom{\beta}{m}}{\binom{n}{m}} \leq 0.443m \tag{6.16}$$

We conclude that our assumption $\overline{\mathcal{Z}}(n,m) \leq 0.443m$ is correct for any n and all $m > 1$. Therefore, for all $m > 2$ and any value of n, $\overline{\mathcal{Z}}(n,m) \leq 0.443m$. □

Proof of Theorem 3.4. According to Eq. (3.3), the $\overline{\mathcal{C}}(n,m)$ can be expressed as

$$\overline{\mathcal{C}}(n,m) = \sum_{i=\mu}^{\nu} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\mathcal{C}\left(\alpha, m-i\right) + \mathcal{C}\left(\beta, i\right) + 1\right] \tag{6.17}$$

Case 1: $m \leq \alpha$ and $m \leq \beta$: Then $\mu = 0$ while $\nu = m$. Let us separate the first two and the last two terms from the summation and evaluate them, our expression becomes

$$\overline{\mathcal{C}}(n,m) = \sum_{i=2}^{m-2} \frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}} \left[\mathcal{C}\left(\alpha, m-i\right) + \mathcal{C}\left(\beta, i\right) + 1\right] + \frac{\binom{\alpha}{m}\binom{\beta}{0}}{\binom{n}{m}} \left[\mathcal{C}\left(\alpha, m\right) + 1\right] +$$
$$\frac{\binom{\alpha}{m-1}\binom{\beta}{1}}{\binom{n}{m}} \left[\mathcal{C}\left(\alpha, m-1\right) + 1\right] + \frac{\binom{\alpha}{1}\binom{\beta}{m-1}}{\binom{n}{m}} \left[\mathcal{C}\left(\beta, m-1\right) + 1\right] +$$

$$\frac{\binom{\alpha}{0}\binom{\beta}{m}}{\binom{n}{m}}\left[\mathcal{C}\left(\beta,m\right)+1\right] \tag{6.18}$$

Following the procedure introduced in the proof of Theorem 3.3, we can substitute $\mathcal{C}\left(\beta,i\right)$ and $\mathcal{C}\left(\alpha,i\right)$ by $1.443i-1$. We then proceed to collect the missing terms from the summation and arrive at the following expression:

$$\overline{\mathcal{C}}(n,m)\leq 1.443m-1+\frac{\binom{\alpha}{m}}{\binom{n}{m}}-1.443\beta\frac{\binom{\alpha}{m-1}}{\binom{n}{m}}-1.443\alpha\frac{\binom{\beta}{m-1}}{\binom{n}{m}}+\frac{\binom{\beta}{m}}{\binom{n}{m}} \tag{6.19}$$

Which can be simplified using the binomial coefficient identity introduced in the proof of Theorem 3.3 to

$$\overline{\mathcal{C}}(n,m)\leq 1.443m-1+\frac{\alpha-1.443m\beta+1-m}{m}\frac{\binom{\alpha}{m-1}}{\binom{n}{m}}+ \frac{\beta-1.443m\alpha+1-m}{m}\frac{\binom{\beta}{m-1}}{\binom{n}{m}} \tag{6.20}$$

Assume that n is even, then $\alpha=\beta=\frac{n}{2}$ and

$$\overline{\mathcal{C}}(n,m)\leq\frac{3}{2}-1+2\frac{\alpha(1-1.443m)+(1-m)}{m}\frac{\binom{\alpha}{m-1}}{\binom{n}{m}}\leq 1.443m-1 \tag{6.21}$$

On the other hand, if n is odd, then $\beta=\alpha-1$, and

$$\overline{\mathcal{C}}(n,m)\leq 1.443m-1+\frac{\beta(1-1.443m)+(2-m)}{m}\frac{\binom{\alpha}{m-1}}{\binom{n}{m}}+ \frac{\alpha(1-1.443m)-m}{m}\frac{\binom{\beta}{m-1}}{\binom{n}{m}}\leq 1.443m-1 \tag{6.22}$$

Therefore, $\overline{\mathcal{C}}(n,m)\leq 1.443m-1$ for any n and all $m>1$.

Case 2: $m\leq\alpha$ and $m>\beta$: Then $\mu=0$ while $\nu=\beta$ and n can only be odd. Further more, $\alpha=m$ and $\beta=m-1$, while $n=2m-1$. Substituting all this in Eq. (3.3), we obtain

$$\overline{\mathcal{C}}(n,m)=\sum_{i=2}^{\beta-1}\frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{m\binom{n}{m}}\left[\mathcal{C}\left(\alpha,m-i\right)+\mathcal{C}\left(\beta,i\right)+1\right]+\frac{\binom{\alpha}{m}\binom{\beta}{0}}{m\binom{n}{m}}\left[\mathcal{C}\left(\alpha,m\right)+1\right]$$

$$+\frac{\binom{\alpha}{m-1}\binom{\beta}{1}}{m\binom{n}{m}}\left[\mathcal{C}\left(\alpha,m-1\right)+1\right]+\frac{\binom{\alpha}{m-\beta}\binom{\beta}{\beta}}{m\binom{n}{m}}\left[\mathcal{C}\left(\alpha,m-\beta\right)+\beta\right] \tag{6.23}$$

From our induction assumption, for all α and $\beta < n$, $\mathcal{C}\left(\alpha,i\right) \leq 1.443i-1$ and $\mathcal{C}\left(\beta,i\right) \leq 1.443i-1$. Thus,

$$\overline{\mathcal{C}}(n,m) \leq 1.443m-1+\frac{\binom{\alpha}{m}}{\binom{n}{m}}-0.443\beta\frac{\binom{\alpha}{m-1}}{\binom{n}{m}}-1.443\beta\frac{\binom{\alpha}{m-\beta}}{\binom{n}{m}} \tag{6.24}$$

From the binomial coefficient property introduced in the proof of Theorem 3.3, we can simplify the above equation to obtain

$$\overline{\mathcal{C}}(n,m) \leq 1.443m-1+\frac{\alpha-0.443\beta+1-m}{m}\frac{\binom{\alpha}{m-1}}{\binom{n}{m}}-1.443\beta\frac{\binom{\alpha}{m-\beta}}{\binom{n}{m}} \tag{6.25}$$

Because $m \leq \alpha$ and $m > \beta$, n can only be odd and $\alpha = m$, while $\beta = \alpha - 1$. Accordingly, we have

$$\begin{aligned}\overline{\mathcal{C}}(n,m) \leq 1.443m-1+\frac{1.443-0.443m}{m}\frac{\binom{\alpha}{m-1}}{\binom{n}{m}}+\\ (1-1.443m)\frac{\binom{\alpha}{m-\beta}}{\binom{n}{m}} \leq 1.443m-1\end{aligned} \tag{6.26}$$

We conclude that, for any n and all $m > 1$, $\overline{\mathcal{C}}(n,m) \leq 1.443m-1$ is correct.

Case 3: $m > \alpha$ and $m > \beta$: Then $\mu = m - \alpha$ while $\nu = \beta$. Therefore, Eq. (3.3) can be written as,

$$\overline{\mathcal{C}}(n,m) = \sum_{i=m-\alpha}^{\beta}\frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}}\left[\mathcal{C}\left(\alpha,m-i\right)+\mathcal{C}\left(\beta,i\right)+1\right] \tag{6.27}$$

Notice that terms with $i = 0$ do not appear in the equation. The smallest value for i is 1. Because $\mathcal{C}\left(\beta,1\right) = \mathcal{C}\left(\alpha,1\right) = 0$, it is true that $\mathcal{C}\left(\beta,1\right) = \mathcal{C}\left(\alpha,1\right) \leq 1.443m-1$; therefore,

$$\begin{aligned}\overline{\mathcal{C}}(n,m) &\leq \sum_{i=m-\alpha}^{\beta-1}\frac{\binom{\alpha}{m-i}\binom{\beta}{i}}{\binom{n}{m}}\left[1.443(m-i)-1+1.443i-1+1\right]+\\ &\quad \frac{\binom{\alpha}{m-\beta}\binom{\beta}{\beta}}{\binom{n}{m}}\left[1.443m-1.443\beta-1+\beta\right]\\ &= 1.433m-1-0.443\beta\frac{\binom{\alpha}{m-\beta}}{\binom{n}{m}} \leq 1.443m-1\end{aligned} \tag{6.28}$$

We conclude that $\overline{C}(n, m) \leq 1.443m - 1$ is correct for any n and all $m > 1$ for all the cases. $\square$

References

[1] N. Amitay, "Resource auction multiple access (RAMA): Efficient method for fast resource assignment in decentralized wireless PCS." *Electronic Letters*, April 9, 1992, V28 N8:799-801.

[2] D. Bertsekas and R. Gallager, *Data Networks*, Second Edition, Prentice-Hall, 1992.

[3] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: a media access protocol for wireless LANs," *Proceedings of ACM SIGCOMM '94*, pp. 212–25, ACM, 1994.

[4] J.I. Capetanakis, "Tree algorithm for packet broadcasting channel," *IEEE Trans. Inform. Theory*, vol.IT-25, pp. 505-515, Sept. 1979.

[5] C. Fullmer and J .J . Garcia-Luna-Aceves, "Floor acquisition multiple access for packet-radio networks," *Proc. ACM SIGCOMM '95*, Cambridge, MA, August 30-September 1, 1995.

[6] C. L. Fullmer and J.J. Garcia-Luna-Aceves, "Solutions to Hidden Terminal Problems in Wireless Networks," *Proc. ACM SIGCOMM 97*, Cannes, France, September 14-18, 1997.

[7] R. Garcés and J .J . Garcia-Luna-Aceves, "Floor acquisition multiple access with collision resolution," *Proc. ACM/IEEE Mobile Computing and Networking '96*, Rye, NY, Nov. 10-12, 1996.

[8] P. Karn, "MACA - a new channel access method for packet radio," *ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, pp. 134–40, ARRL, 1990.

[9] M. J . Karol and I. Chih-Lin, "A protocol for fast resource assignment in wireless PCS," *IEEE Transactions on Vehicular Technology*, vol. 43, no. 3, pp. 727-32, IEEE, 1994.

[10] M. J. Karol, Liu Zhao, K. Y. Eng, "Distributed-queuing request update multiple access (DQRUMA) for wireless packet (ATM) networks," *Proc. IEEE ICC '95*, Seattle, WA, USA, June 18-21, 1995.

[11] L. Kleinrock and F. A. Tobagi, "Packet switching in radio channels: Part I - carrier sense multiple-access modes and their throughput-delay characteristics," *IEEE Trans. Commun.*, vol. COM-23, no. 12, pp. 1400–1416, 1975.

[12] W. F. Lo and H. T. Mouftah, "Carrier sense multiple access with collision detection for radio channels," in *IEEE 13th International Communications and Energy Conference*, pp. 244–47, IEEE, 1984.

[13] G. S. Sidhu, R. F. Andrews, and A. B. Oppenheimer, *Inside AppleTalk, Second Edition.* Addison-Wesley Publishing Company, Inc., 1990.

[14] F. A. Tobagi and L. Kleinrock, "Packet switching in radio channels: Part II - the hidden terminal problem in carrier sense multiple-access modes and the busy-tone solution," *IEEE Trans. Commun.*, vol. COM-23, no. 12, pp. 1417–1433, 1975.

[15] W. Xu and G. Campbell, "A distributed queuing random access protocol for a broadcast channel," *Proc. ACM SIGCOMM'93*, San Francisco, USA, 13–17 Sept. 1993.

```
Variable Definitions
    CD = Carrier Detected
    $T_{PROP}$ = Maximum channel propagation delay
    $T_{PROC}$ = Processing time for carrier detection
    $Burst$ = Number of packets to send in a burst
    $Stack$ = Stack to keep tract of backoffs
    $N$ = Total number of stations
    $ID$ = Station ID number
    $LowID$ = Lowest ID number allow to send RTS
    $HiID$ = Highest ID number allow to send RTS

Procedure START()
Begin
    $LowID \longleftarrow 1$
    $HiID \longleftarrow N$
    $Stack \longleftarrow$ NULL
    Timer $\longleftarrow 2 \times T_{PROP}$
    While($\overline{CD} \wedge$ Timer not expired) wait
    If (CD) Then call REMOTE($2 \times T_{PROP} + T_{PROC}$)
    Else call PASSIVE()
End

Procedure PASSIVE()
Begin
    While($\overline{CD} \wedge$ No Local Packet) wait
    If (CD) Then call REMOTE($2 \times T_{PROP} + T_{PROC}$)
    Else Begin
        If ($LowID \leq ID \leq HiID$) Then
            call RTS($2 \times T_{PROP} + T_{TR} + T_{PROC}$)
        Else call BACKOFF()
    End
End

Procedure RTS($T_\sigma i$)
Begin
    Transmit RTS Packet
    Timer $\longleftarrow T_\sigma$
    While($\overline{CD} \wedge$ Timer not expired) wait
    If (Timer expired)
    Then Begin
        call Update($LowID$, $HiID$, $Stack$)
        If ($LowID \leq ID \leq HiID$) Then
            call RTS($2 \times T_{PROP} + T_{TR} + T_{PROC}$)
        Else call BACKOFF()
    End
    Else Begin
        Receive Packet
        DO CASE of (received packet type)
        Begin
            CTS: call XMIT()
            Error:
                call Update($LowID$, $HiID$, $Stack$)
                If ($LowID \leq ID \leq HiID$) Then
                    call RTS($2 \times T_{PROP} + T_{TR} + T_{PROC}$)
                Else call BACKOFF()
        End
    End
End

Procedure BACKOFF()
Begin
    Timer $\longleftarrow T_{RTS}$
    While($\overline{CD} \wedge$ Timer not expired) wait
    If (CD) Then call REMOTE($2 \times T_{PROP} + T_{PROC}$)
    Else Begin
        If ($LowID \leq ID \leq HiID$) Then
            call RTS($2 \times T_{PROP} + T_{TR} + T_{PROC}$)
        Else call BACKOFF()
    End
End

Procedure XMIT()
Begin
    Burst $\longleftarrow$ maximum burst
    While ((Burst > 0) $\wedge$ Local Packet)
    Do Begin
        Transmit Data Packet
        Burst $\longleftarrow$ Burst - 1
    End
    Timer $\longleftarrow T_{PROP}$
    While (Timer not expired) wait
    call Update($LowID$, $HiID$, $Stack$)
    If (Local Packet)
    Then Begin
        If ($LowID \leq ID \leq HiID$) Then
            call RTS($2 \times T_{PROP} + T_{TR} + T_{PROC}$)
        Else call BACKOFF()
    End
    Else call PASSIVE()
End

Procedure REMOTE($T_\sigma$)
Begin
    Timer $\longleftarrow T_\sigma$
    While($\overline{CD} \wedge$ Timer not expired) wait
    If (Timer expired)
    Then Begin
        call Update($LowID$, $HiID$, $Stack$)
        If (Local Packet)
        Then Begin
            If ($LowID \leq ID \leq HiID$) Then
                call RTS($2 \times T_{PROP} + T_{TR} + T_{PROC}$)
            Else call BACKOFF()
        End
        Else call PASSIVE()
    End
    Else Begin
        Receive Packet
        DO CASE of (received packet type)
        Begin
            RTS: call REMOTE($2 \times T_{PROP} + T_{PROC}$)
            CTS:
                if (Destination ID = Local ID) Then
                    Transmit CTS Packet
                call REMOTE($2 \times T_{PROP} + T_{PROC}$)
            DATA:
                if (Destination ID = Local ID)
                Then pass packet to upper layer
                call REMOTE($T_{PROP} + T_{PROC}$)
            ERROR: call REMOTE($2 \times T_{PROP} + T_{PROC}$)
        End
    End
End
```

Figure 1. CARMA Specification

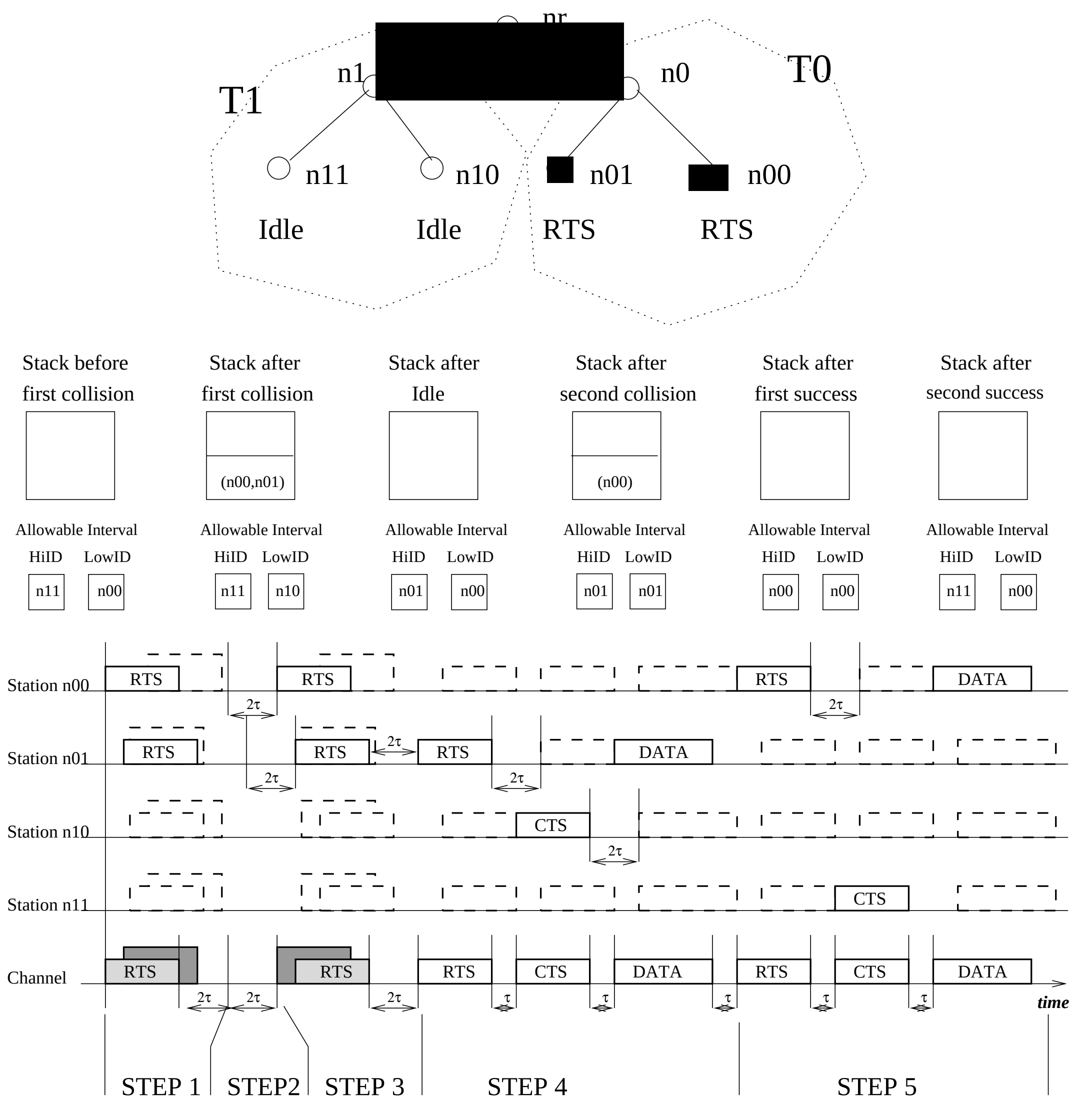

Figure 2. Transmission period and tree structure to solve the collisions for a system with $n = 4$ stations out of which $m = 2$ are requesting the floor. $\mathcal{C}(4,2) = 2$, $\mathcal{S}(4,2) = 1$ and $\mathcal{Z}(4,2) = 1$.

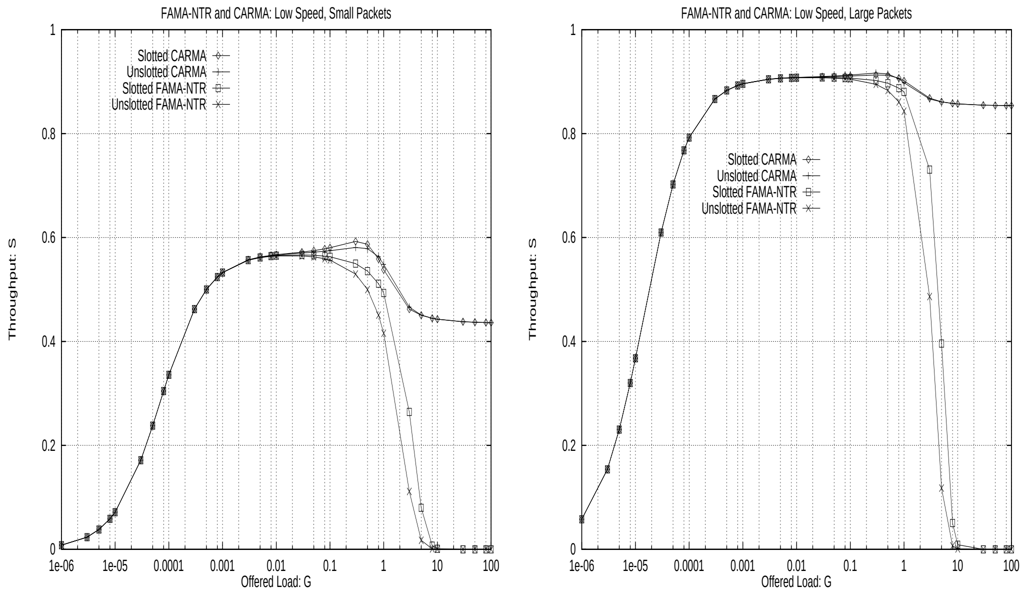

Figure 3. Throughput of FAMA-NTR and CARMA for low-speed network.

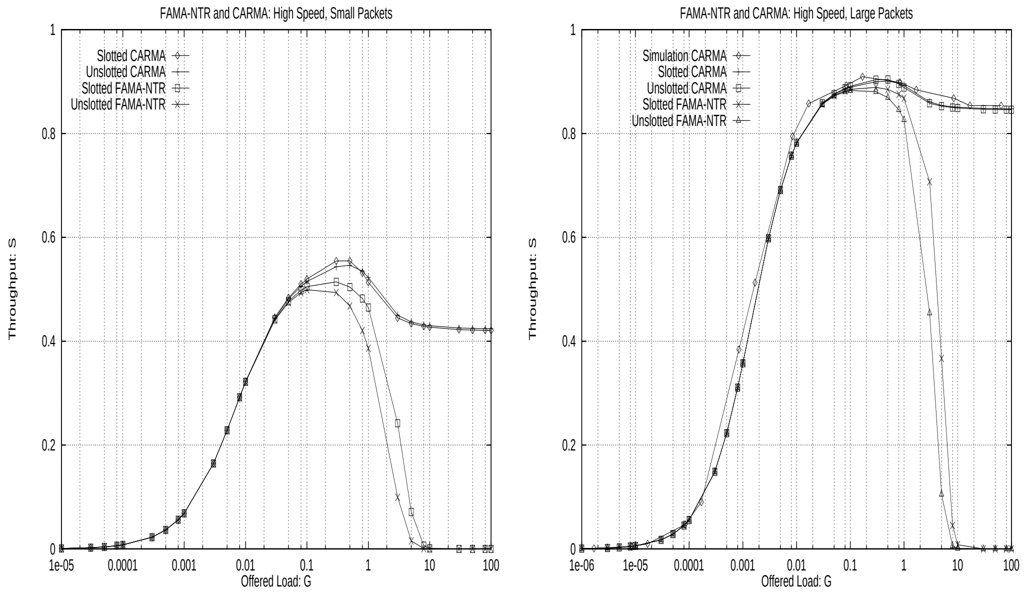

Figure 4. Throughput of FAMA-NTR and CARMA for high-speed network.

Table 1
Rules for the average number of steps.

Rule 1:	$\mathcal{C}$(n,0) = 0	**Rule 4:**	$\mathcal{S}$(n,0) = 0	**Rule 7:**	$\mathcal{Z}$(n,0) = 1
Rule 2:	$\mathcal{C}$(n,1) = 0	**Rule 5:**	$\mathcal{S}$(n,m) = m	**Rule 8:**	$\mathcal{Z}$(n,1) = 0
Rule 3:	$\mathcal{C}$(n,n) = n-1	**Rule 6:**	$\mathcal{S}$(n,n) = n	**Rule 9:**	$\mathcal{Z}$(n,n) = 0

Table 2
Number of steps required to solve all possible permutations for a tree with $n = 4$ and $m = 2$.

n_{11}	n_{10}	n_{01}	n_{00}	C Steps	S Steps	Z Steps
idle	idle	RTS	RTS	2	2	1
idle	RTS	idle	RTS	1	2	0
idle	RTS	RTS	idle	1	2	0
RTS	idle	idle	RTS	1	2	0
RTS	idle	RTS	idle	1	2	0
RTS	RTS	idle	idle	2	2	1

Table 3
Throughput equations for FAMA-NTR and CARMA protocols

Protocol	*Unslotted Version*	*Slotted Version*
FAMA [5]	$S \geq \frac{ae^{-G}}{(a+b+1)e^{-G}+b+4+\frac{1}{G}}$	$S \geq \frac{aGe^{-G}}{(aG+bG+2G-b-3)e^{-G}+b+4}$
CARMA	$S \geq \frac{(e^{-G}(G-1)-G)a}{A' \cdot e^{-G}+B'}$; $A'=(a+3.433b+6.732)G-a-3b-5$ $B'=(-a-3.433b-6.732)G-\frac{1}{G}+b$	$S \geq \frac{(G \cdot e^{-G}-1)aG}{A' \cdot e^{-G}+B'}$; $A'=(a+3.433b+5.299)G^2+(0.433b+1.299)G+1-b$ $B'=(-a-3.433b-5.299)G+b-2$

Table 4
Protocol variables for low-speed networks (9600 bps) and high-speed networks (1 Mbps) with two types of data packets, small (424 bits) or large (3200 bits). The channel delay $\tau = 5.4\mu$s, while the control packets are 160 bits long.

Network Speed	Packet Size	δ	$a = \frac{\delta}{\tau}$	$b = \frac{\gamma}{\tau}$
9600 bps	424 bits	44166.7 μs	8179.0	3086.4
9600 bps	3200 bits	333333.3 μs	61728.4	3086.4
1 Mbps	424 bits	424 μs	78.5	29.6
1 Mbps	3200 bits	3200 μs	592.6	29.6

Table 5
Initial conditions for the average number of steps.

$\mathcal{Z}(1,0) = 1$	$\mathcal{Z}(3,2) = \frac{1}{3} \leq 0.886$	$\mathcal{C}(1,0) = 0$	$\mathcal{C}(3,2) = \frac{4}{3} \leq 1.886$
$\mathcal{Z}(1,1) = 0$	$\mathcal{Z}(3,3) = 0 \leq 1.299$	$\mathcal{C}(1,1) = 0$	$\mathcal{C}(3,3) = 2$
$\mathcal{Z}(2,0) = 1$	$\mathcal{Z}(4,0) = 1$	$\mathcal{C}(2,0) = 0$	$\mathcal{C}(4,0) = 0$
$\mathcal{Z}(2,1) = 0$	$\mathcal{Z}(4,1) = 0$	$\mathcal{C}(2,1) = 0$	$\mathcal{C}(4,1) = 0$
$\mathcal{Z}(2,2) = 0 \leq 0.886$	$\mathcal{Z}(4,2) = \frac{1}{3} \leq 0.886$	$\mathcal{C}(2,2) = 1 \leq 1.886$	$\mathcal{C}(4,2) = \frac{4}{3} \leq 1.886$
$\mathcal{Z}(3,0) = 1$	$\mathcal{Z}(4,3) = 0 \leq 1.299$	$\mathcal{C}(3,0) = 0$	$\mathcal{C}(4,3) = 2 \leq 3.329$
$\mathcal{Z}(3,1) = 0$	$\mathcal{Z}(4,4) = 0 \leq 1.732$	$\mathcal{C}(3,1) = 0$	$\mathcal{C}(4,4) = 3 \leq 4.772$